Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 5 (2011) 697–704

Procedia Computer Science

The 8th International Conference on Mobile Web Information Systems (MobiWIS)

A Platform for Interactive Location-Based Services

W. Ait-Cheik-Bihi*, M. Bakhouya, A. Nait-Sidi-Moh, J. Gaber, M. Wack

Systèmes et Transports Laboratory, Université de Technologie de Belfort-Montbéliard, Rue Thierry Mieg, Belfort 90010, France

Abstract

The rapid expansion of wireless technologies and their interactions with wired networks together with wide spread of Global Navigation Satellite Systems (GNSS) infrastructures allow today the development of location-based services (LBS). Furthermore, the evolution of service oriented infrastructures enables information sharing between applications over the network. In this paper, a distributed interactive platform, called TransportML, for LBS interaction and cooperation is presented. It is an extended XML platform based on Services Oriented Architecture (SOA) principles to recognize, to monitor, and to collect service information and description over the network. A prototype is developed, in the context of intelligent transportation systems (ITS), to allow sharing information collected by each service and then acquiring more information about the driving context. This information helps drivers making appropriate decisions and, therefore, increases and improves road safety. The prototype is tested in urban and highway areas and results are reported to show the benefit of using this platform for sharing road information as well as services interaction.

Keywords: Location-based services, GNSS, SOA, Inter-vehicle communication, ITS applications.

1. Introduction

Wireless communication technologies associated with GNSS allow now individuals and professionals to access LBS over Internet wherever they are and whenever they want. LBS are information services that are accessible with mobile devices through the mobile network and utilizing the ability to make use of the location of the mobile device [7]. There exist a broad range of different location based services. One of the most important applications of LBS is the ability to locate a person who is either unaware of his/her exact location or is not able to reveal it because of an emergency situation (e.g., vehicle break down). With the exact location automatically transferred to the emergency services, the assistance can be provided as quickly as possible. However, this happens without considering the availability of other information on the road network. In other words, each geographical area benefits from a set of wide range of local services, which are usually provided by several local entities, such as snow clearance, road works, public transportation, waste collection, fire fighting or emergency rescue. Despite each service fulfils its own assigned task, they often suffer from a lack of communication between each other. Indeed, each service collects road-related information that can be useful to other services. For example, information collected (e.g. snow

* *Corresponding author: Tel.: +33-(0)3-84583953; fax: +33-(0)3-84583342.*
E-mail address: wafaa.ait-cheik-bihi@utbm.fr

Selection and/or peer-review under responsibility of Prof. Elhadi Shakshuki and Prof. Muhammad Younas.
doi:10.1016/j.procs.2011.07.091

clearance status, congestion, road work) can be used for the itinerary computation, and hence the assistance will be provided quickly and efficiently.

To achieve this objective, a TransportML middleware is developed to allow several entities (e.g., road operators) to share their information using interactive Web services in order to fulfil common tasks. Web services are self-contained, modular units of application logic that provide business functionality to other applications via an Internet connection [3,18]. It can be considered as a standard interface between LBS, which expose their data according to SOA principles. More precisely, TransportML allows Web services to automatically interact without a predefined collaboration scenario. Indeed, the scenario is determined at execution time, by using the TransportML-compliant semantic description of the services. As a consequence, TransportML is highly dynamic and extensible, allowing new services to join the community and benefit from its communication medium. This effort is a part of our work in the EU FP7 project, ASSET (Advanced Safety and Driver Support for Essential Road Transport)*, with the main objective is to specify and develop a collaboration platform for local entities' services interactions. Tools and guidelines are also provided to service providers; so that they can expose the information they possess and join the collaboration platform.

The rest of the paper is organized as follows. In section 2, we briefly survey existing work related to services collaboration and SOA principles. Section 3 introduces TransportML platform. In section 4, we present the testbed prototype, experiments, and report the obtained results. Conclusion and further work are presented in section 5.

2. Background and related work

In recent years, with the pervasive availability of wireless communication technologies and handheld devices, considerable research effort has been done in the area of inter-vehicle communication. The objective is to increase the driver safety and comfort by relaying required information from vehicle to vehicle. For example, future equipped vehicles are expected to anticipate and avoid possible collisions, navigate the quickest route to their destination, and identify the nearest available parking slot [4]. Furthermore, the emergence of GNSS infrastructures has stimulated extensive research in the development of LBS and applications. For developing such services, different infrastructure components are required: mobile devices, communication networks, positioning components, and service providers. Devices, such as Laptops, PDA, and cell phones, are used to request these services and get back results. The mobile network is another component required to transfer the users' requests from mobile devices to the services providers and send the requested information back the users. Positioning components are used to determine the location of the users in order to customize their required information. Services providers are located in a fixed infrastructure (i.e., Internet) and offer a number of different services to the users. However, a platform is required to facilitate the development and deployment of LBS to provide users pervasive accesses to these services as well hide details about technological infrastructures.

Several platforms have been developed for general purpose mobile applications [11-17] and for LBS [7-10] as well. In the best of our knowledge none of them address the development of interoperable LBS for transportation roads in order to increase safety. In addition to the interoperability issue addressed by these platforms, LBS for transport roads introduce new requirements, such as the need of service interactions/collaborations and the large amount of existing information that must be handled efficiently by these platforms.

SOA has emerged in the past few years for the development and integration of systems over Internet where functionalities are packaged as independent interoperable services. Furthermore, SOA's strength lies in the loose coupling of services with operating systems, programming languages, and other underlying technologies. In other words, SOA splits and packages features in distinct entities – services – that developers make accessible on a network so that they can be reused and combined to develop new services over the Web. A Web service is a W3C standard defined as a software system designed to support interoperable machine-to-machine interaction over a network [5,6]. Web services use a standard XML format to exchange information and they are not tied to a particular operating system or any programming language.

The Web service architecture is composed of three main entities: the service provider, the service broker and the service requester. The service provider hosts a Web service which can be invoked by the service requester (i.e.

* http://www.project-asset.com/

client) through requests formatted in SOAP XML-based format [1]. To satisfy the need for Web service discovery, the service provider should register its service to the service broker. The service broker is actually a service directory (usually UDDI [5]), which can be interrogated by services requesters to find services that can satisfy their requests. However, this architecture cannot be directly used to develop transport road-based LBS and support their interactions.

In this paper, a TransportML platform based on SOA principles is introduced to enable interaction and collaboration of road-related services. It can be directly accessible by mobile users using any technology allowing an Internet connection. For instance, GPRS technology has been identified to be the most suitable for location-based applications deployment [7].

3. TransportML architecture

The overall architecture of the TransportML middleware is presented in Fig. 1 (a). This architecture allows providing value-added services (itineraries computation in the current version), resulting from the collaboration between existing services maintained by different transport road entities. In this context, a service is a standard way for companies, associations or other organizations, such as municipalities and local authorities, to expose their internal information.

TransportML is composed of three components, registry, request handler, and orchestration components as depicted in Fig. 1 (b). The TML-formatted user request (see Fig. 1 (c)) is transmitted to the orchestration component, which checks for services registered into registry component, and determines services (via services registry) to be invoked to satisfy the request. The orchestration component sends then sequentially the TML document to all invoked service providers. The orchestration component sends then the resulting TML document, which includes advised and unadvised areas, to the itinerary service to compute the best suitable itinerary. Once the orchestration component receives the TML document with the computed itinerary it sends it back to the request handler component, which in turn provides the end user by the computed itinerary.

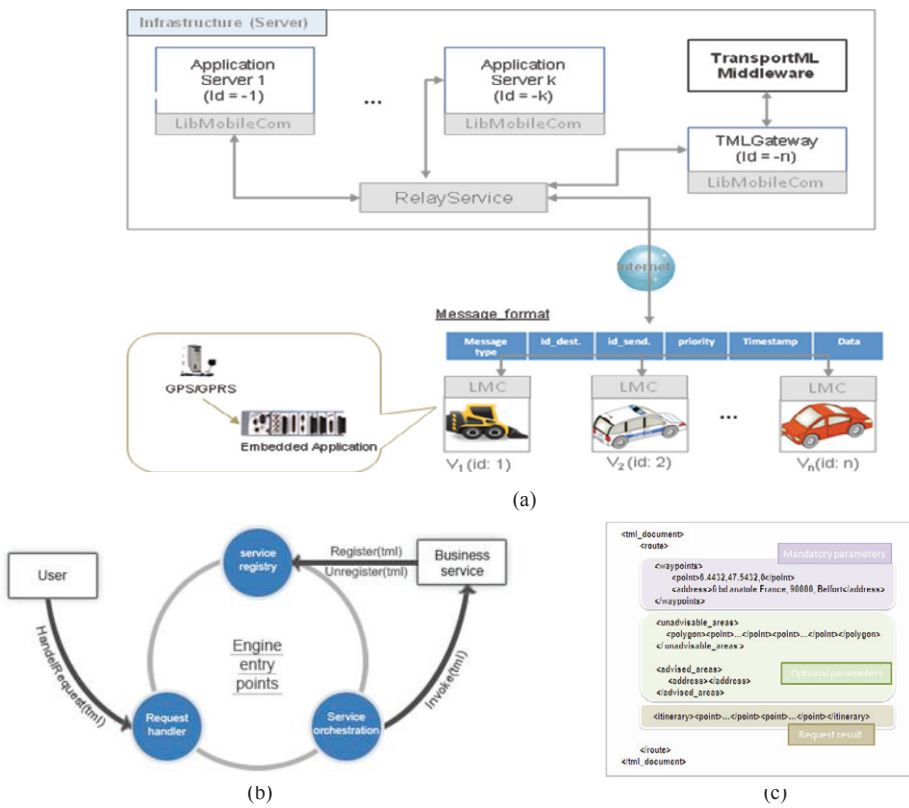


Fig. 1. (a)TransportML architecture, (b) TransportML components, (c) TML document

To allow the communication between the vehicles and the TransportML platform, a communication layer is developed and integrated. It is composed of a Relay server and a LibMobileCom (LMC) library for V2I2V communications as shown in Fig. 1 (a). The LMC is a Java Communication library used by the entities (i.e., servers, vehicles) to exchange messages. The Relay server registers all participants and forwards messages back and forth. TMLGateway receives messages with "TML" message type, parses the upload data under the TMLDocument format, and then sends this document to TransportML platform. Once this later handles the request, the response is sent back to TMLGateway as a TMLDocument, which is then sent via the communication protocol based on the LMC to the requester. The main advantage of adding the TML Gateway server is to avoid the overhead caused by XML encapsulation.

The LibMobileCom library will take care of formatting the messages as illustrated in Fig. 1 (a). Each message includes six fields, the message type in three letters (e.g. "GPS" for exchanging GPS coordinates) in order to know the message content, destination identifier, sender identifier, priority (equals to 1 if the message can't be dropped when the destination or the network is not available and 0 in the other case), timestamp, and data (the message content). The relay server only parses the destination ID and the priority, and then forwards the message to the destination. All messages are buffered with FIFO priority in case of not availability of the destination entity; otherwise they will be immediately submitted to their destinations for processing after being parsed by LibMobileCom. In other words, only messages having low priority could be drooped to avoid buffer overflow. Furthermore, the protocol overhead is kept to a minimum to save bandwidth and decrease the communication latency. This wireless communication API is available for public usage*.

Fig. 1 (c) shows the message format, called TMLDocument, exchanged between all TransportML entities, which consists of XML document having a **<tml_document>** root element. This root element has one child element corresponding to the type of request: **<route>,** which in our case specifies that the user is asking for a route. We also refer to this element as a section of the TML document. The <route> section contains several children corresponding either to parameters (mandatory or not) or request result. In the selected example, the **<waypoints>** element contains the route request mandatory parameters. At least two waypoints must be provided: the start point and the end point. The **<advised_areas>** and **<inadvisable_areas>** elements correspond to optional parameters which can be filled in by third-party services, before itinerary computation, in order to provide added value. Finally, the last child element, **<itinerary>,** contains the response to user's request. It's worth noting that TransportML in its current version is dedicated to itineraries computation using a specific request format, which doesn't require a semantic engine. For other value-added services with additional capabilities, semantic features will be incorporated.

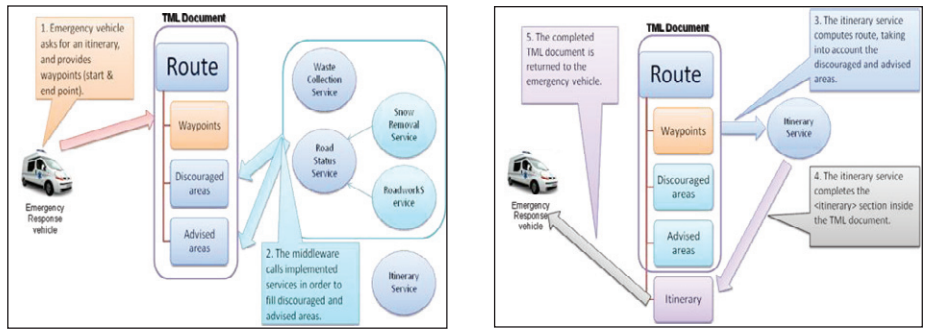
4. Testbed prototype and results

A prototype is developed to demonstrate the feasibility of TransportML platform and highlight the platform capabilities to coordinate distributed LBS. The prototype of TransportML platform is developed using Java EE5 and runs on Glassfish Application server. The development was made using Netbeans 6 IDE, which includes SOA development tools. Netbeans 6 IDE can be used to develop both Web service and Web service clients using JAX-WS API. These Web services can then be deployed on the Glassfish application server for testing and evaluation.

4.1. Testbed description

Based on road-related knowledge gathered from different local services that are road status service, snow clearing service, and geofencing service, for test purpose we focused on advanced itinerary computation type of requests. The geofencing service provides information about granted or denied areas to vehicles in accordance with their registered characteristics such as weights, speed, height, type (trucks or cars), and type of good transported. It provides information concerning zones wherein traffic can be difficult and congestion may appear. Snow clearance service provides information regarding snow clearance status, i.e., clearing roads progress. Roadwork service provides information regarding on-going work zones or damaged road surface.

* www.gsem.fr/V2V/



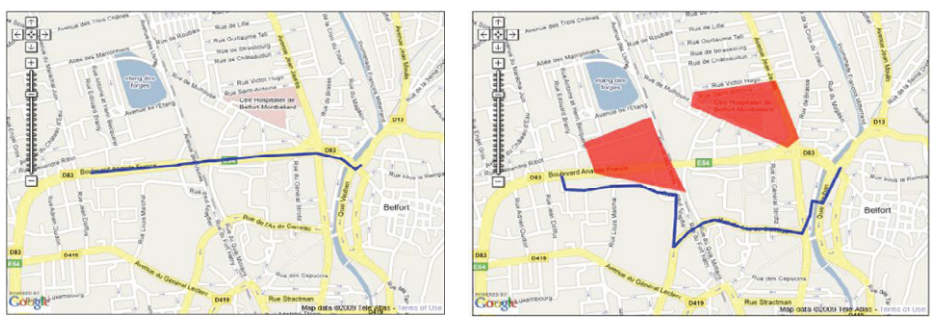
(a) (b)

Fig. 2. Itinerary computation process, (a) received request and calls services, (b) compute itinerary and send back response

The itinerary computation service computes the fastest route between two geographical locations using information provided by the services described above into two steps. First the vehicle characteristics and waypoints are defined and sent to the platform. The query is then handled by invoking the required services to define the appropriate itinerary. In other words, whenever TransportML platform receives a client request (e.g. “I am looking for a fast itinerary from point A to point B.”), it is supported through the request handler and the service orchestration engine discovers and selects services to be invoked. The service orchestration engine will then call those services in order to get the TML document submitted with all the required parameter items filled in. This first step of the scenario is depicted in Fig. 2 (a). Once all these services have processed the different items, a TML document with <inadvisable_area> and <advised_areas> parameter items filled in is obtained. Then, the second step of the scenario starts by calling an itinerary service to compute the required itinerary which is sent out to the requester as illustrated in Fig. 2 (b).

4.2. Results and discussion

The key point in our test scenarios is that computing an optimized itinerary service by making use of inadvisable and advised areas information, provided by other services, is an added-value produced by the platform and enabled by the inter-service collaboration. In other words, this added-value service guarantees optimal path duration to reach the destination. It’s worth noting that reducing this duration is very important in the case where the requester entity is actually an emergency vehicle.



(a) (b)

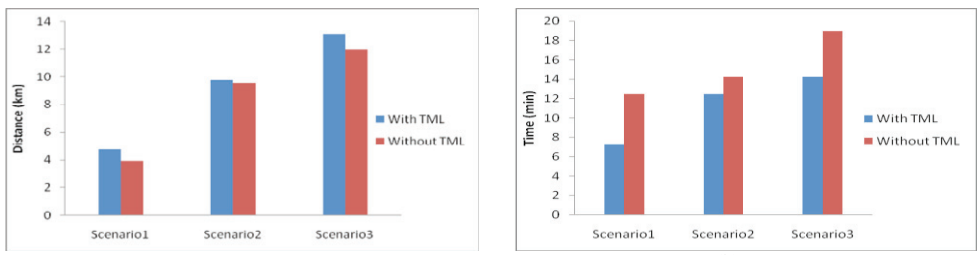
Fig. 3. Itinerary computed (a) without and (b) when using TransportML

A Web service client application to visualize the response received by the user is developed using Google maps API. A TransportML Java library that supports KML conversion was developed for that purpose. The experiments have been done in Belfort city both in urban and highway environments. The objective is to study the impact of inter-services collaboration on the efficiency of emergency intervention. The itinerary visualized on the screenshot illustrated in Fig. 3 (a) is an itinerary computed without considering inadvisable or advised areas. Fig. 3 (b) shows the itinerary obtained with the help of TransportML; inadvisable areas, provided by the other services, are taken into

account to compute an advised itinerary.

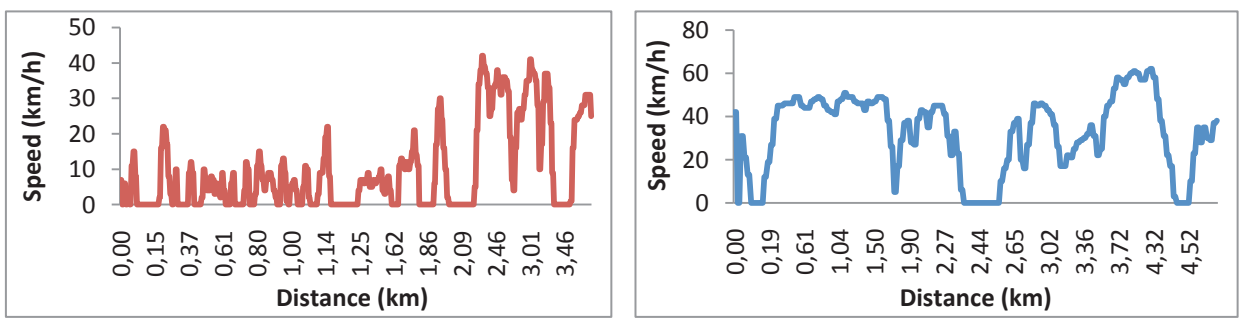
To highlight the impact of TransportML, we have compared the distance and the amount of time needed to go from the departure to the destination location in the both cases: with and without using TransportML. Three scenarios are considered in the experiments, a scenario1 with one service, scenario2 with two services, and scenario3 with three services. For each scenario, the departure and arrival points are different. The results illustrated in Fig. 4 (a) show that, in all scenarios, when using TransportML, the distance travelled is longer than the distance travelled when TransportML is not used. However, Fig. 4 (b) shows that the time required for the same travelling path is lower when TransportML is used.

In fact, by using TransportML, roads that are not cleared yet from snow, inadvisable areas, or work-zones are avoided, and therefore, the amount of time needed to reach a destination is shorter even if the distance is longer. In other words, without using TransportML, the time is longer since the vehicles, when traversing inadvisable areas, undergo a slowdown (traffic jam or limited speed on road sections due to work-zones ...). The obtained results show that the average travelling time is around 27% less when using TransportML.



(a) (b)

Fig. 4. Distance (a) and time (b) for the computed itinerary with and without using TransportML



(a) (b)

Fig. 5. The speed versus distance, (a) without TransportML (b) with TransportML

During the experimentations, the vehicle speed and the locations are collected every 3 seconds. We have measured driving patterns by collecting the vehicle's speed and position from GPS Datalogger to get the time and the distance travelled. Fig. 5 illustrates the vehicle speed variation during the distance travelled with and without using TranportML. Fig. 5 (a) shows that, without using TransportML, the vehicle slowdowns to become fully stationary for some period of time. In other words, in inadvisable area, traffic jam starts to appear at the beginning of this area, and vehicle's speed increases/decreases slowly, which leads to an increase of acceleration/decelerations (approximately on 2 km). This period corresponds to the congestion (the time to skip the queue length) in the zone defined as an inadvisable area. However, when using TransportML (Fig. 5 (b)), the vehicle speed increases but decreases some certain period of time. The vehicles move at relatively constant speed (except in stop/lights locations), and no congestion behaviour was detected. During these experiments, we have concluded that, traffic congestion mainly caused by road capacity, traffic incidents, work zones, weather conditions, should be avoided to decrease emergency travelling time (see Fig. 4).

We finally evaluate the services call time and the communication time. The results are presented in Fig. 6. When

TransportML is used, which means that several services might be invoked, the time cost, needed to call the involved ones to get back the TML document filled with advised and inadvisable areas, increases slightly despite the services are sequentially called. The average communication time also increases slightly with the number of services and the size of the TML document response increases as well. In fact, the time required to send the TML document is proportional to its size and to the available bandwidth.

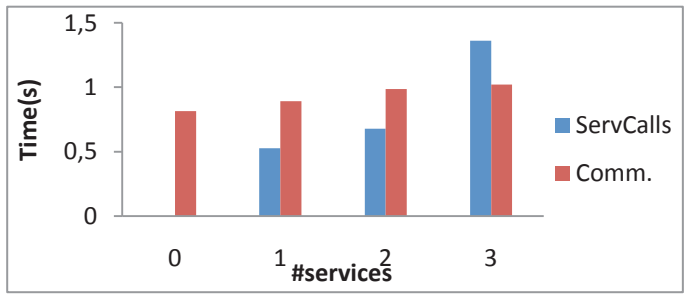
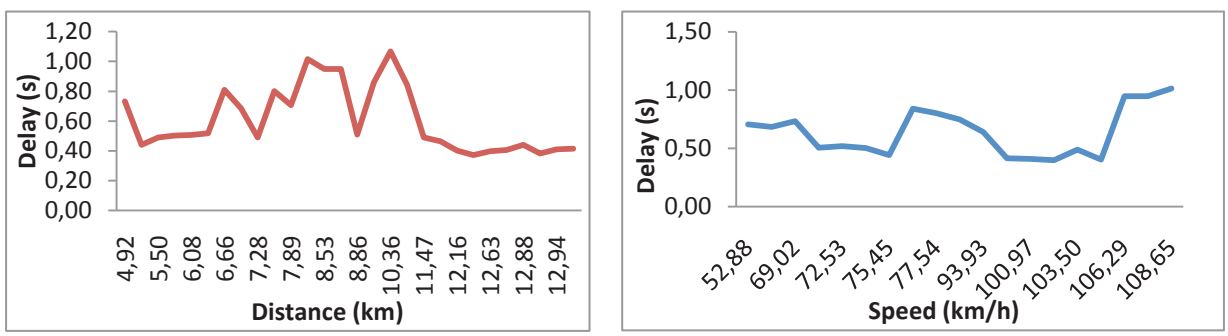


Fig. 6. Communication and call services times

To show the sensitivity of 3G communication technology, the delay to send out information (e.g., speed, position, direction) from the in-vehicle embedded computer to the infrastructure (i.e., local server) was measured. The obtained results show that 3G technology is neither sensitive to distance, from the vehicle to the server locations as shown in Fig. 7 (a), nor to the vehicle's speed (Fig. 7 (b)). The average communication time is approximately 0.6s.



(a) (b)

Fig. 7. The wireless communication delay versus (a) V2I distance and (b) the vehicle's speed

5. Conclusion and future work

In this paper, the TransportML platform which allows automatic road-services interaction and collaboration is presented. TransportML is highly dynamic, allowing new services to join the community and benefit from its communication medium. The platform is highly extensible because no predefined scenario is required at development time. A prototype was developed and experiments have been conducted in both urban and highway environments, the results are reported to show the effectiveness and the impact of inter-services collaboration for reducing emergency rescue time.

The obtained results showed that the average travelling time from the emergency location to the incident location is around 27% less when using TransportML, with a small additional overhead. This decrease in travelling time is due to the computed itinerary that bypasses inadvisable areas wherein congestion was detected during the experiments. We have also evaluated the additional overhead incurred by TransportML, mainly the services call, and the wired and wireless communication. Because of space limitation, TML documents parsing, services discovery and selection, interaction scenario elaboration and services invocation are not presented in this paper. The comparison of TransportML with other platforms is an ongoing work. The scalability of the platform for large scale usage together with additional experiments will be considered. Further work will address the integration of additional features and services such as eCall service [19] with aftermarket nomadic devices (e.g., personal

navigation devices, Smartphones).

Acknowledgements

This work is supported by the FP7-SST EU project ASSET (Advanced Safety and Driver Support for Essential Road Transport, 2008-2011, http://www.project-asset.com/).

References

1. M. Gudgin, M. Hadley, M. Mendelsohn et al., 'SOAP Version 1.2 Part 2: Adjuncts', W3C Recommendation 24th June 2003 (URL: http://www.w3.org/TR/2003/REC-soap12-part2–20030624/).

2. Bellwood T., Clément L., Ehnebuske D., Hately A., Hondo M., Husband Y.L., Januszewski K., Lee S., McKee B., Munter J., and von Riegen C.. "UDDI Version 3.0", July 2002.

3. Dustdar S., & Schreiner W. (2005). A Survey on Web services Composition, Int. Journal on Web and Grid Services, 1(1), 1-30, Inderscience Enterprises Ltd.

4. K. Dar, M. Bakhouya, J. Gaber, M. Wack, P. Lorenz, Wireless Communication Technologies for ITS Applications, IEEE Comm. Magunications Magazine, 48: 5. 156-162, 2010.

5. Akram M. S, Medjahed B., & Bouguettaya A. (2003). Supporting Dynamic Changes in Web Service Environments. In Proceedings of the International Conference on Service Oriented Computing, LNCS Volume 2910/2003, pp. 319-334, 978-3-540-20681-1.

6. Wu C., & Chang E. (2005). State-of-the-art Web Services Architectural Styles. Proceedings of the Third IEEE European Conference on Web Services (ECOWS), Vaxjo, Sweden. Available at http://wscc.info/p51561/files/paper54.pdf

7. Virrantaus, K., Markkula, J., Garmash, A., Terziyan, Y.V., 2001. Developing GIS-Supported Location-Based Services. In: Proc. of WGIS'2001 – First International Workshop on Web Geographical Information Systems., Kyoto, Japan. , 423–432.

8. H. K. Y. Leung, I. Burcea, and H.-A. Jacobsen. Modeling location-based services with subject spaces. In Proceedings of the 2003 conference of the Centre for Advanced Studies conference on Collaborative research, pp. 171–181. IBM Press, 2003.

9. T. Hadig and J. Roth. Proximity services with the nimbus framework. In IADIS International Conference Applied Computing, pp. 437–444, Lisbon, Portugal, Mars 2004. IADIS Press.

10. I. Burcea and H.-A. Jacobsen. L-Topss - push-oriented Location-based Services. In 4th VLDB Workshop on Technologies for E-Services (TES'03), number 2819 in Lecture Notes in Computer Science, pp. 131–142, Humboldt University, Berlin, Germany, September 2003. Elsevier.

11. L. Capra. Mobile Computing Middleware for Context-aware Applications. In Proceedings of the 24th international conference on Software engineering, pp. 723–724. ACM Press, 2002.

12. R. Bagrodia, T. Phan, and R. Guy. A Scalable, Distributed Middleware Service Architecture to Support Mobile Internet Applications. Wireless Networks, 9(4):311–320, 2003.

13. G. Cugola and H.-A. Jacobsen. Using Publish/Subscribe Middleware for Mobile Systems. SIGMOBILE Mobile Computing and Communications Review, 6(4):25–33, 2002.

14. G. Cugola, E. D. Nitto, and A. Fuggetta. The JEDI Event-based Infrastructure and its Application to the Development of the OPSS WFMS. IEEE Transactions on Software Engineering, 2001(27):827–850, 2001.

15. N. Davies, A. Friday, S. P. Wade, and G. S. Blair. L2imbo: a Distributed Systems Platform for Mobile Computing. Mob. Netw. Appl., 3(2):143–156, 1998.

16. C. Mascolo, l. Capra, S. Zachariadis, and W. Emmerich. XMIDDLE: A Data-sharing Middleware for Mobile Computing. Journal on Personal and Wireless Communication, April 2002.

17. G. P. Picco, A. L. Murphy, and G.-C. Roman. LIME: Linda Meets Mobility. Conference on Software Engineering, pp. 368–377, 1999.

18. M Bakhouya, J Gaber. Service Composition Approaches for Ubiquitous and Pervasive Computing Environments: A Survey. In Agent Systems in Electronic Business, Edited by:Eldon Li and Soe-Tsyr Yuan. Information Science Reference/IGI Publishing, pp. 323-350, 2007.

19. W. Ait-Cheik-Bihi, A. Chariette, M. Bakhouya, A. Nait-Sidi-Moh, J. Gaber, and M. Wack. An In-vehicle Emergency Call Platform for Efficient Road. In the proceedings of European ITS Congress, 6-9 June, Lyon, France, 2011.